Unit 2: HTML Site Building

Learn the elements of HTML.

Site:	School of Computing and Information Systems
Course:	Computer Science 266: Introduction to Web Programming (Revision 3)
Book:	Unit 2: HTML Site Building
Printed by:	Hector Barquero
Date:	Saturday, 25 May 2024, 9:38 AM MDT

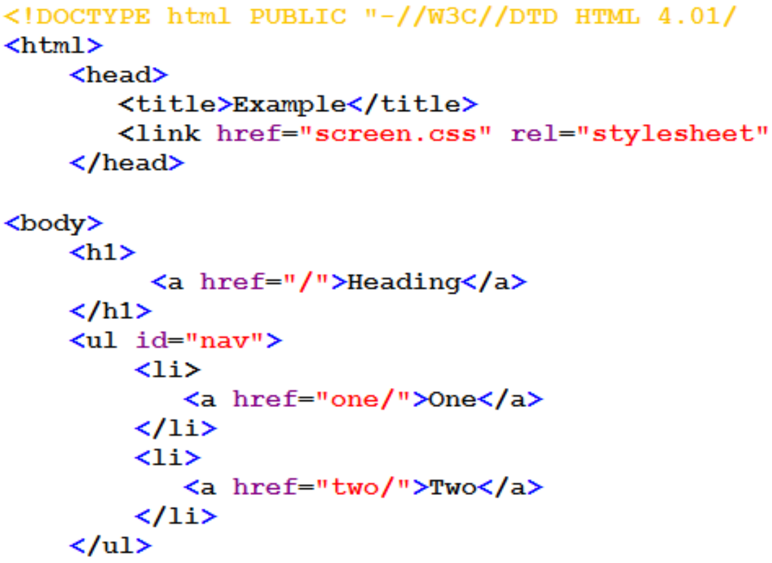
Table of contents

Study Guide Unit 2 HTML Site Building

Overview

☞ Unit 2 is worth 15% of your portfolio grade

Podcast - Introduction to Unit 2



```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01/
<html>
    <head>
       <title>Example</title>
       <link href="screen.css" rel="stylesheet"
    </head>

<body>
    <h1>
         <a href="/">Heading</a>
    </h1>
    <ul id="nav">
        <li>
           <a href="one/">One</a>
        </li>
        <li>
           <a href="two/">Two</a>
        </li>
    </ul>
```

This unit is about writing Hypertext Mark-up Language (HTML), which is a skill you will need and continue to develop throughout the course. HTML is very simple to learn and quite hard to master. To help you get the hang of it, we are giving you two very badly written HTML pages to modify and turn into the foundation of your site, which will be based on the work you did for Unit 1. We also ask you to create a third page all by yourself. Like everything on this course, your pages will be based on the themes, purposes, personas and scenarios identified in Unit 1.

The badly written HTML code provides the first lesson: amazingly, most desktop browsers will display this fairly well, despite its many weaknesses (mobile browsers and ones that are fussier about syntax may have problems, though). The useful first lesson about HTML is that it is possible to make some quite terrible mistakes and for your code to still do what it is meant to do . . . mostly. However, it is very hard to predict what different browsers will do with such code and extremely hard to maintain it.

We are going to try to help you to learn to make code that works well for everyone and can be maintained and extended easily. However, our main interest here is not to help you to be a web designer: you cannot program in JavaScript within a web browser unless you already know how to write HTML (and, to a slightly lesser extent,CSS—coming in the next unit).

Study Guide Unit 2 HTML Site Building

Learning Outcomes

When you have completed this unit, you should be able to write well-structured, easily maintained, standards-compliant, accessible HTML code.

Study Guide Unit 2 HTML Site Building

The Problem

Very important: You should not begin this task until you have completed Unit 1, including the reflective learning diary.

We have prepared two terribly written HTML files: sample1.html and sample2.html, that you should download to a directory you create for that purpose on your own computer. The object is to add your own content to fit the needs identified in Unit 1, and to make it better: accessible, maintainable, standards-compliant HTML that addresses some of the needs you identified in Unit 1. You will also need to create a new page from scratch, linked to these.

> ☞ For this unit, we don't care at all what it looks like and *require* that you avoid using CSS, JavaScript, or dynamic content of any kind, as well as avoiding any HTML that is concerned with the appearance of the page—no fonts, colours, alignments, or other features that relate to the look of the page: there should be **no visual mark-up**.

The emphasis in this exercise is that the HTML has to be **well structured, very accessible, very maintainable, very standards compliant with no deprecated code and no poor coding practices.** You will need to follow tutorials and references in order to understand what that means. We require

- a minimum of **three** HTML pages complying with at least XHTML 1.0 s
- at least one **image.**
- **hyperlinks** between all the pages (using relative URLs).
- at least one **hyperlink to an external website**.
- sufficient text to require the use of at least two heading styles
- at least one **list** (ordered or unordered).
- at least one **<div> tag** (preferably more), with a specified name, to identify a section of a page (of no particular value yet, but needed later on). It would be best if this were something that you see the need to reformat later to change its appearance, position, or behaviour, for example a sidebar floated on one side of the page.
- at least one **<span> tag** (preferably more), with a specified name, to identify a section of text within a paragraph or other block-level element (again, no immediate value—for use later). It would be best if this were something that you see the need to reformat later to change its appearance, position, or behaviour, for example, a change in typeface or font.
- a **table.** (*Special note regarding tables*: tables are only ever to be used for tabular data, with meaningful rows and columns of data, *never* to lay out things on the page—such use is common but strongly deprecated because of accessibility issues).

min xhtml 1.0 or better
image min = 1
relative urls for hyperlinks
external hyperlinks min = 1
min 2 heading styles
min 1 list
minimum of one div tag, ideally more
at least one span tag, ideally more
a table - not for design, but for data
a form that mails to the author

- a **form**, the contents of which are mailed to the author (you).

Top

It is really important that your pages **communicate** well: the purpose of almost every page is, first and foremost, to convey information and knowledge. Programming is also, at least partly, an exercise in communication. Do not skimp on the content; make sure you check your spellings; try to avoid breaking standard grammar rules (unless it helps with the communication—if so, don't forget to tell us that it is intentional in your learning diary). Also remember that clear communication can be aided by good structure, sensible use of headings, appropriately sized paragraphs and pages, and meaningful images (with proper consideration of accessibility for those who cannot see them).

In addition to the pages themselves, you should provide (in your learning diary) a **critique** of the templates you have been given, explaining why they were poorly written and how you have improved them.

In your reflective learning diary, you should also provide an **explanation** of how your pages fit with the needs of the personas and scenarios you identified in Unit 1. If, as a result of this, you need to make changes to Unit 1 work, you should explain what changes were made and why you made them. Do not simply overwrite the originals—submit corrections as new files.

if u need to make changes, dont overwrite-- add new. unfortunate bc it adds so much file bloat.

Things we will look for in your pages include

- well structured, standards-compliant HTML code with no deprecated tags, attributes, or uses of HTML
- effective use of the scenarios and personas from Unit 1, with clear rationales and appropriate design decisions based on how you expect the pages to be used and who you expect to use them.
- a broad range of HTML tags and attributes used, but only when appropriate, not for the sake of it. Examples might include heading styles, tables, hypertext links, anchors, lists (ordered, unordered, definition), divs, spans, text formatting (descriptive, not related to appearance).
- accessibility – suitable for users with different browsers and different abilities/disabilities.
- usability – bearing in mind personas and scenarios from Unit 1 and taking into account that different people may access your site with markedly different browsers (including mobile devices).
- effective navigation between pages (bearing in mind usability and accessibility constraints).
- effective organization of files: images, for example, should be saved in a separate directory (later there will be other subdirectories for different kinds of files).
- author name and date of last modification on each page.
- images sized appropriately for the Web, with useful *alt* attributes for those unable to see them.
- clear, effective communication that is appropriate to the personas and scenarios you have identified in Unit 1.

key marking points check list

Unit 2 HTML Site Building

Process Guide

To do this exercise, you will have to learn HTML. In common with the approach taken throughout this course, although we give you a brief introduction to the topic and some suggested reading, we expect that you should make use of web-based resources to do this yourself. We will support you in the process, give feedback if needed, and encourage discussion with others on the course, but we will not give you a set of steps to follow nor tell you exactly what to do, as that method of learning doesn't work very well for most people.

Read the introduction below then take your pick of tutorials and references using either those provided or others that you or fellow course-mates have found. If you find good resources, share them on the Landing.

Please use the course discussion forums on the Landing to discuss issues and problems as they arise, whether in process or technology. If you can help others with their problems, please do so: it helps them and it helps you (there are few better ways to learn than to teach someone else), and you can use the help you give as evidence of successfully meeting the learning outcomes for the course if you wish.

If and when you find useful resources, please add these to the course bookmarks on the Landing. Please include sufficient information to help others understand not only what the resources are about, but also how the resources you have found will help them in their learning. Don't forget to provide suitable tags to make it easier for people to find them. Again, this all counts as evidence that you may use to help gain more marks, so it's a way of helping yourself while helping others.

Unit 2 HTML Site Building

Editing HTML Files

There are several ways you can create, edit, and save HTML files. WYSIWYG (what you see is what you get) tools such as Dreamweaver defeat the purpose of this unit, which is to learn how HTML works “under the hood.” Notepad, which comes with Windows, is a popular choice. You simply save your file with the .html extension.

Mac OS X: How to Set Up TextEdit as an HTML or Plain Text Editor

You can do a search on the Internet for more information if necessary.

Unit 2 HTML Site Building

Submitting Your Work

You should save your code to the directory on [SCIS server provided for this purpose], and provide a link to this in your reflective learning diary.

> ☞**Important:** create a zipped-up version of the code (no need for the images) and save it as an attachment to your diary entry. You must do this at the end of each unit. This will help to provide us with a clearer image of how your site has developed over the course, what you have learned, and how you have developed your skills.

You will receive no formal feedback on these pages at this point but, if you are not sure whether you are working along the right lines, please ask your tutor for a quick appraisal. He or she may suggest strengths or weaknesses for you to work on, but note that he or she will spend no more than 15 minutes on this task, so the feedback will not go into great depth. You can sometimes supplement this by helping and being helped by others on the course: feel free to explore (and comment on) the work of other students, if available. You may get ideas by doing this, and it will help you to gauge roughly where your work lies in relation to others.

(Almost) Everything You Need to Know about HTML in a Few Short Paragraphs

HTML is the language of the web. It was created by Tim Berners-Lee as a subset of SGML. As a standard, it is managed by the W3 Consortium, or W3C for short. At the time of writing, there are two main current versions: HTML 4.01 and XHTML. The two are very similar, but XHTML is more rigorously defined as a form of XML, and this rigour makes it more suitable for our purposes here. Either works in all modern browsers in almost exactly the same way, though there are some slight differences that can catch the unwary developer.

At the time of writing, HTML 5 is an emerging set of standards that looks likely to replace both, but has not yet been fully ratified, and there are various incomplete, partial, and conflicting versions. For now, it is a safer bet to learn about HTML 4.01 or, better still, XHTML. We will accept both and will also be happy with HTML 5, but whichever standard you use, you should be explicit and consistent about which one you are using and avoid anything that is described as deprecated, even if the standard allows it in principle.

Unit 2 HTML Site Building

Tags

HTML, whatever the version you use, is simply composed of plain text, some of which has a special meaning to web browsers. These special parts are called tags and are distinguished from other text by being enclosed in angle brackets (e.g. <p> or <h1> or <table>). They are just plain text that could be written using any text editor. The web browser reads the page (technically, it *parses* it) and, depending on the tags it finds, renders the page accordingly.

While all web pages consist of only plain text, sometimes they may use that text to describe how to display other forms of content like pictures, movies, Flash animations, or music. These are always separate files, separately loaded by the web browser, and never part of the HTML itself. The HTML tags *point* to such files and, generally, give the browser a bit of information about what to do with them, such as where an image should appear in relation to the other content of the page, alternative text to display if the browser cannot show the content, and so on.

A particularly important tag creates the hyperlinks used to connect one page with others (technically, these are anchor tags, for historical reasons). We will use this anchor tag to explain the basics of how HTML works.

The vast majority of tags come in two parts—an opening tag and a closing tag. The closing tag is the same as the opening tag but with a forward slash (/) before it. For example, for our hyperlink, we start with an <a> tag and close with a </a> tag. The tag affects anything that is added between the opening and closing tag. In the case of our hyperlink, the default behaviour is to underline any text or image that appears between <a> and </a> and make it “clickable,” sending you somewhere else. But how does the browser know where to go? We need to specify the URL (the web address) that the user of our page will be sent to. This is specified using an attribute. An attribute adds extra information to a tag to let the browser know how it is intended to behave.

All tags can include further attributes. Attributes are things that modify the tag so that it can behave differently. They consist of an attribute name, an equal sign (=), and a value. The value is always enclosed in quotation marks. Common attributes might look something like *id="something" or alt="some alternative text"*.

In the case of our <a> tag, we need to tell it to where we want to hyperlink it. For example, <a href="http://www.google.com">Go to Google</a> will send us to Google. Notice that the attribute has a name (href) that is set to be equal to a value (http://www.google.com), which is enclosed in quotation marks. This tells the browser where to go when our hyperlink is selected. Notice too that the opening and closing tags enclose some text. That is the text that will form the hyperlink on which people will be able to click. You can find out a bit more about hyperlinks on this page.

Most tags behave much the same way, with an opening and closing tag enclosing some content (usually text), and all can take zero or more attributes to modify their behaviour. For example, <p>This will create a simple paragraph</p> and <h1>This will appear as a heading</h1>, typically shown by the browser in a large font on a separate line. <em>This text will appear to be emphasized</em>, typically in italic lettering while <strong>this will usually appear in bold text</strong>.

Almost all tags can include other attributes: for instance, <p id="myparagraph"> attaches an identifier that you can use elsewhere in your code to make it behave or look differently. Note that attribute values must always be enclosed in quotation marks. Attributes are only applied to opening tags and never applied to closing tags: </p id="myparagraph"> would be wrong, for example.

Unit 2 HTML Site Building

Keeping Content and Presentation Separate

Notice how the tags are describing to the browser *what* to do, not specifying *how* it should be done. It is up to the browser to decide how to render the results, and different browsers can render the same HTML in different ways. For instance, a browser that uses text-to-audio to read the text of a page (a screen reader) for accessibility may change the pitch or intonation when it comes across an <em> tag, while another browser may show it in italic font, while another may show it in a different colour.

Older and poorly written HTML sometimes mixes up the logical and the visual presentation and, in the past, there were even tags that encouraged you to do this—<b> for bold, <i> for italic, <font> to specify a font, and <center> to centre text, for example. This kind of tag is *strongly* deprecated nowadays and should not be used. HTML should be about the structure of the content, not what it looks like. The job of presentation is now left to a different and far more powerful and flexible technology, CSS, of which more in the next unit.

We mentioned that the vast majority of tags have closing tags associated with them. However, for a few tags it makes little sense to create a closing tag—for example, line breaks, images, and tags that embed other files would usually enclose nothing else so it would be pointless (though it would not be syntactically wrong) to add closing tags. In such cases, you simply add a forward-slash within the tag itself. For instance, <br /> inserts a line break; <hr /> inserts a horizontal rule. As both of these have more to do with appearance than structure, it is generally better to avoid them.

However, there are some such tags that really do matter—images are a particularly common case. <img src="mypicture.jpg" alt="My picture" /> inserts an image called “mypicture.jpg” on the page. Notice that the image tag, like all tags, can have multiple attributes. In the case of the <img> tag it is, of course, necessary to provide a name for the image so that the browser knows which image to display, but note that it is also very important that you include an 'alt' attribute to provide alternative text when either the browser will not display the image (some people turn them off; some browsers are text-only) or the person reading the page may not be able to see it. In other words, it is a tag that is intended to improve accessibility of the page. Although the browser will happily display an image if the tag has no 'alt' attribute set, in many countries you might be in breach of the law in providing inaccessible content on your web page. Again, remember that HTML is describing the content, not specifying how it should be displayed. In the case of an image, that description generally needs to be a bit fuller than a simple tag alone.

Many tags are only meaningful in a particular context defined by other tags. For example, a table row only makes sense in the context of a table, while table data (a cell) only makes sense if it appears on a table row.

Study Guide Unit 2 HTML Site Building

Nesting and Lists

COMPLETED

You must be careful that your tags are properly nested, or you may get unpredictable results, you will find your code hard to maintain, and very strict HTML-compliant browsers may refuse to display anything at all. For example, <p><em>Hello world</em></p> is good, while <p><em>Hello world</p></em> is not.

Another very important example of such nesting is when building lists. You can define an unordered list (<ul>) or an ordered list (<ol>), for example, which contain list items (<li>). List items outside of a list do not make a lot of sense. You can find out more about lists here.

Another notable example of nesting is that any tag that relates to something displayed on a web page must appear inside <body> </body> tags. Similarly, a page title (usually displayed in the browser's title bar) should only ever appear inside the <head> </head> tags.

Study Guide Unit 2 HTML Site Building

Standards and Browsers

There are dozens of tags available to be used. If you use a tag that the browser doesn’t understand, it will just ignore it. You are perfectly welcome to make up your own tags if you like, but there is seldom any point as there will not be any browsers that know what to do with them, and they will just be ignored. This makes HTML very resilient to failure, but also allows us to do some quite interesting tricks, including applying our own special mark-up that we can use in other ways, perhaps to add styling, embed different content, or define a database. That is far beyond what is covered in this course, however.

HTML standards are defined by the W3 consortium at http://w3.org (which also offers good tutorials on HTML, CSS, and JavaScript). Few, if any, browsers conform to all standards perfectly, and there can even be differences in the same browser running on different platforms (Mac, Windows, Linux, iPhone, Blackberry, etc.), which means that not all HTML works in the same way for every user. Microsoft is particularly notorious for ignoring or breaking standards although their more recent browsers are mostly a little better than their old ones.

Mainly for this reason, we strongly recommend that you use other browsers, especially for testing your code: Firefox is a particularly good choice as it runs on most platforms, has many tools available to help with coding, and is free. But others are just as good or better: Apple Safari (Mac and Windows), Opera (most platforms), Google Chrome (most platforms), or Konqueror (Linux and Unix) are all good choices and are available at no cost. Firefox, Chrome, and Konqueror are open source as well as free. Whatever browser you mainly use, the more of them that you can test your code with the better. In this course, code that only runs on a single browser will be marked down. Another advantage of using other browsers is that most of them are much faster, more secure, and more reliable than Microsoft's IE.

Of course, this brief overview is nothing like everything you need to know! Please use the provided tutorials to teach yourself more, or, if you like, find others and share them with your course-mates via the bookmarks of our Landing group.

If you'd like to try some really basic HTML, you might consider this exercise that covers the minimum needed.

Unit 2 HTML Site Building

Indicative Grading Criteria

For this basic exercise we would allocate marks roughly as follows (bearing in mind that this is presented as part of the portfolio at the end of the course, and you will have opportunities to work on it again before then):

50%: HTML structure, maintainability, standards compliance, accessibility, comprehensiveness—including use of heading styles, text emphases, images, multiple pages, hyperlinks, forms, tables.

15%: Accuracy and depth of critique

30%: Linking with personas and scenario

5%: Effective communication

Grade	Criteria
A	• very well-structured, standards-compliant code with no visual mark-up • a large set of tags properly used • accessible to users with a wide range of disabilities (including visual, motor and cognitive) • well organized, with any use of the work of others properly cited • design decisions and content for the pages fully justified in the context of personas and scenarios, with a very clear rationale for every decision you make • critique of the template code identifies all of the errors and provides improvements that would make them perfectly formed, syntactically correct, and with all the necessary elements and attributes of a good HTML document • excellent communication

B	• well-structured, standards-compliant code with no visual mark-up • a wide range of tags properly used • accessible to users with a wide range of disabilities (including visual, motor and cognitive) • well organized, with any use of the work of others properly cited • design decisions and content for the pages fully justified in the context of personas and scenarios, with a clear rationale for every decision you make • critique of the template code identifies almost all of the errors and provides improvements to make them well-formed, syntactically correct, and with all the necessary elements and attributes of a good HTML document • well communicated
C	• correctly structured, standards-compliant code • little or no visual mark-up • a good range of tags used, covering the basic needs of the assignment • accessible for most users with disabilities • properly cited • justified in the context of personas and scenarios but not always naturally following from them • small coding errors • weaknesses in effectiveness of communication • critique identifies most errors
D	• minor weaknesses in structure and standards compliance • occasional use of visual mark-up • enough tags to satisfy the requirements but not always used effectively—for example, failure to use headings where they would be needed • accessible to blind people • adequate critique identifying some errors, but with some corrections in the code not matched to the critique • poor communication (e.g., spelling errors, poor layout, weak structure) • the occasional error in HTML or content (e.g., improper use of absolute URLs, missing attributes, wrongly sized images, etc.)

Unit 2 HTML Site Building

Resources

- Self-teach HTML tutorials, relevant standards pages, online tools to test compliance and clean up code (e.g., HTMLTidy)
- Self-study multiple-choice questions/missing word/etc., exercises that give drill and practice on identifying poorly written code
- WCAG guidelines for accessibility
- W3C references

☞ If you find other useful resources, please add links to them on the Landing so that others may benefit. You may use such activities as supporting evidence of having met the learning outcomes.

Basic Tutorials

http://www.w3schools.com/html/default.asp

http://www.tizag.com/beginnerT/

http://htmldog.com/guides/htmlbeginner/

http://www.tizag.com/htmlT/

References

http://w3c.org
The source of all web standards. If you want the definitive reference that defines the standard, come here. Not always easy reading, however.

http://www.w3.org/TR/WCAG20/
Web content accessibility guidelines (Version 2.0). Your code should comply with these.

Tools

Web Developer Toolbar for Firefox—fantastic tool, a must-have toolkit for any web developer.